TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 1, February 2020, pp. 166∼173
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i1.14105

Integrity verification for an optimized cloud architecture

Iqbal Ahmed, Fahim Irfan Alam
Department of Computer Science & Engineering, University of Chittagong, Chittagong, Bangladesh

Article Info

Article history:

Received Sep 14, 2019
Revised Dec 6, 2019
Accepted Dec 21, 2019

Keywords:

Advanced encryption standard (AES)
Cloud architecture
Integrity verification
Secret sharing scheme
Third party auditor (TPA)

ABSTRACT

Cloud computing has significantly benefited today's environment of IT industry through its mobility, sustainability, security, cost savings and several other important features. The risk of data loss on cloud in case of dealing with hardware and software is relatively small. However, the issue of data security becomes imminent when we are storing personal data on the cloud which is not transparent to users. In this paper, we introduce a new entity in terms of a virtual machine that provides services and assurance beyond service level agreement (SLA). In the proposed model, a role of data handling and security is assured with association of third party auditor (TPA) by the virtual machine. We further demonstrate the applied technique for encryption, decryption and integrity verification modules. We also upgrade the entropy of the advanced encryption standard (AES) with a variant of secret sharing scheme in the environment of cloud simulator.

Corresponding Author:

Iqbal Ahmed,
Department of Computer Science and Engineering,
University of Chittagong,
Chittagong, Bangladesh.
Email: iqbal.ahmed@cu.ac.bd

1. INTRODUCTION

Cloud computing has become a tremendous means of technology in the IT industry by offering several important benefits. It allows us to set up what is essentially a virtual office and gives us the flexibility of connecting to us anywhere, any time. Instead of purchasing and installing expensive equipment, we can use the resources of our cloud computing service provider to reduce the cost of system upgrades such as new hardware and software. On cloud, it is also easier to scale up or scale down our storage needs quickly to suit our situation, therefore allowing flexibility as our requirements change. In case of unexpected crisis, such as power failure, having the data stored in the cloud guarantees that it is properly backed up and minimize the loss of productivity. Using cloud services is also cost effective as users pay for the services that are being used by them either fully or partially, e.g. pay per use model [1–3].

Cloud computing provides the interaction between clients and the server with different layers. Software as a Service (SaaS) [4] provides different applications as a service but user is not allowed to make any changes in the service, for example, google Apps, sales force.com. Platform as a Service (PaaS) [5] provides a platform so that customer can run their application on the cloud platform. However, user cannot make any changes to the platform as their control are limited to the application only. Infrastructure as a Service (IaaS) [6] provides the set of hardware and software, storage devices, CPU cycles and other components that make it easier to use the above services. In this way, users gain full control over the structure as they can modify or update it according to their requirements [1, 7, 8].

Cloud computing provides three deployment models: private, public and hybrid (as show in Figure 1). Public clouds are the most popular way of cloud deployment in which the cloud resources are owned and operated by a third-party service provider and can be shared between different business enterprises and organizations [9]. Clients access services and manage their individual accounts using a web browser, e.g. Microsoft Azure. A private cloud consists of computing resources used exclusively by one client and cannot be shared with external parties. It is completely managed by internal policy schemes which ensure that the hardware and software are dedicated solely to the client's organization. Hybrid clouds combine on-premises private clouds infrastructure with public clouds so that organizations can acquire the advantages of both by moving the data and applications between private and public clouds for better flexibility and new deployment options [1, 10].

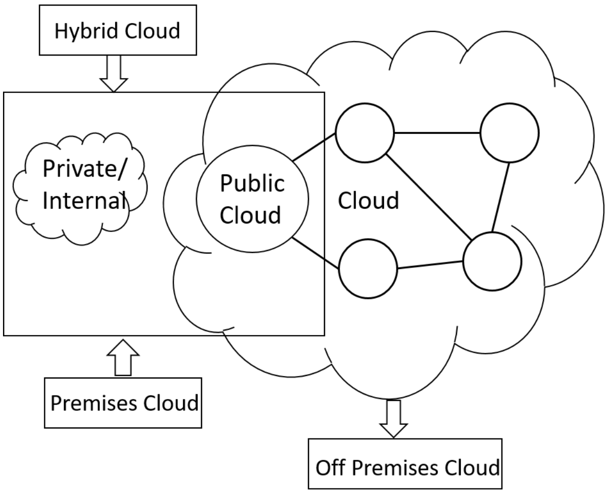


Figure 1. Cloud computing deployment model

In a cloud architecture, it is very important to ensure data integrity which guarantees that data can only be accessed and modified by appropriate authorized entities. It is a possibility that sometimes, the cloud service providers may discard or alter the data without saving in the storage space or keeping backups or replicas. It is also possible that cloud service providers may hide any accidental data loss and claim that the clients' data are still stored in the cloud storage [11–14]. As a result, data integrity verification at untrusted servers remains as one of the biggest concerns with cloud data storage [15].

The proposed work is going to identify a simplified methodology to reconstruct a secret that is distributed using Shamir's Secret Sharing Scheme [16, 17], and to use the derived results to investigate implications on Advanced Encryption Standard (AES) [8, 18] and examine the computation integrity of the data that is outsourced from the local storage to decentralized servers on the clouds. In reality, the users no longer have physical possession of possibly large size of outsourced data which makes the integrity protection in the cloud computing a very challenging and potentially formidable task.

We studied different algorithms that are efficiently working in cloud communication decentralized environment. But there is a great deal of chances of improvement. Specifically, we can focus on the following:

(a) In decentralized cloud communication, security is not assured with each participating virtual machines [19].
(b) Consistency of data or compromise in the length of data is another major security issue in cloud communication.
(c) Communication overhead is increased when we have to communicate with number of machines for retrieving the data.
(d) Storing the short password on the cloud is challenging.
(e) Due to low security, trust factor is also decreased.

2. PROBLEM FORMULATION

We analyzed and concluded that when data is encrypted with stronger algorithms like AES [20] and others, AES works in 64 bit, 128 bit, 256 bit key length and provides a stronger encryption [8]. The limitations of AES are only the lack of implementation and key management [10]. That makes it vulnerable to brute force attack. Secret sharing scheme is working in multicast group where a number of users are sharing sensitive information and trying to perform the complete information confidential [17, 21]. Encryption and multicast group communication can be used together to improve the performance of both. Integrity of the cloud data is to be enhanced by mentioned technique.

3. BACKGROUND

3.1. Secret sharing algorithm

Secret sharing scheme (SSS) is used when we want to divide our key into number of persons and we require all of the persons to be together at one place to get the unlock [16, 22]. There is variant of secret sharing called Threshold secret sharing integers [17]. We present the list of steps of the secret sharing scheme in Algorithm 1.

Algorithm 1 Secret Sharing Scheme (SSS) Algorithm

/ Procedure */*

1. D selects a field F and v different elements $u_1, u_2, \ldots, u_v \neq 0$ of F and interacts with u_i to i^{th} party ($i = 1, 2, 3, \ldots, v$). The secret S ia an element of F.

2. D secretly and randomly selects $t - 1$ elements $p_2, \ldots, p_t$ of the field F.

3. D calculate the shares

$$w_i = S \oplus \bigoplus_{j=1}^{t-1} p_{j+1} \odot u_i^j (i = 1, 2, \ldots, v), \tag{1}$$

4. When different parties want to compute secret, they interpolate and compute S using Lagrange interpolation mechanism:

$$S = \bigoplus_{j=1}^{t} w_{i_j} \odot \bigodot_{\substack{k=1 \\ k\neq j}}^{t} (u_{i_k} \ominus u_{i_j})^{-1} \odot u_{i_k} \tag{2}$$

3.2. Limitations of SSS

There is no provision applied for verifying integrity at each participating virtual machine. If threshold value is very low then security can be breached, if threshold value is high then computational complexity of overall system is will be increased. In classical secret sharing, shares are not produced randomly with time. No default encryption scheme is applied with the algorithm; it has to be embedded with any of the encryption technique [17, 21, 23].

3.3. Entropy calculation

Low entropy means that our source probably is not really random. In particular, if we have a random variable X that takes on values $x_1, x_2, \ldots, x_n$ with probabilities $p(x_1), p(x_2), \ldots, p(x_n)$ respectively, then the entropy of X is:

$$H(X) = -\sum_{i=1} np(x_i)logp(x_i) \tag{3}$$

This value is maximized when all of the probabilities are the same. If we have 2^n different symbols, that maximum value will be n bits of entropy per symbol. That is the theoretical maximum level of entropy that we can get.

4. PROPOSED MODEL

AES is a widely adopted, highly efficient and secure encryption algorithm, used in cloud communication encryption [8, 11, 18]. In our proposed model, we applied AES technique and modified it to secure the cloud communication. We generalized the steps of AES encryption algorithm as follows:

(a) Generate the number of rounds from the cipher key length
(b) Initialize an array with data blocks i.e plain text
(c) Add the first round key to initial state of array
(d) Implement nine rounds of state manipulation
(e) Perform last and second last round of state manipulation
(f) Use final state array as encrypted data Set (cipher text).

We will now present two set of procedures that will elaborate the steps of our proposed model. At first, we will present the system setup process, followed by introducing the modified integrity enhancement module.

4.1. Phase 1: System setup

In the first phase, we present the system setup steps that contain the process of how to convert an input message to the corresponding hexadecimal form and how we apply polynomial interpolation process to generate number of shared for each virtual machine [24, 25]. The system setup steps are listed in Algorithm 2.

Algorithm 2 System Setup

```
/* Procedure */
1. Choose $N$ virtual machines for communication
2. Calculate threshold value $T_p = \frac{(N+1)}{2}$
3. Input a secret message $M$ of $L$ bytes
4. Convert the plaintext message $M$ into hexadecimal
5. Convert the hexadecimal secret into integer
6. Apply polynomial interpolation mechanism to generate no of shares for each machine
7. Polynomial interpolation $(M, T_p, N)$
8. Store all points of polynomial
```

4.2. Phase 2: Integrity enhancement module

In phase 2, we represent the integrity enhancement module of our proposed system for each participating virtual machine. We applied the proposed module to enhance the adaptability of default encryption techniques. We present the basic steps of the integrity enhancement module in Algorithm 3.

Algorithm 3 Integrity Enhancement Module

```
/* Procedure */
1. Calculate authenticator key $H_r$
2. Divide the message $M$ into chunks of 16 bytes
3. Apply random function $R_f$ to generate nonce, a one time unique random number
4. Calculate MAC $h_r(m)$
5. Calculate Q,
        $$Q = \frac{L}{16} \quad (4)$$
6. Store individual message chunks $\{C_1, C_2, \ldots, C_q\}$ of 16 bytes into the array Q[]
7. Construct a polynomial function of $\{C_1, C_2, \ldots, C_q\}$
8. Evaluate the polynomial $f(x)$ at $r$, additional key $h_r(m) = f(r)$
9. Apply $h_r$ key on the message and generate cipher text of desired length
```

5. PROPOSED CLIENT-SERVER ARCHITECTURE

In this section, we will discuss the client-server architecture that we have integrated into our proposed model. The proposed client-server architecture is presented in Figure 2.

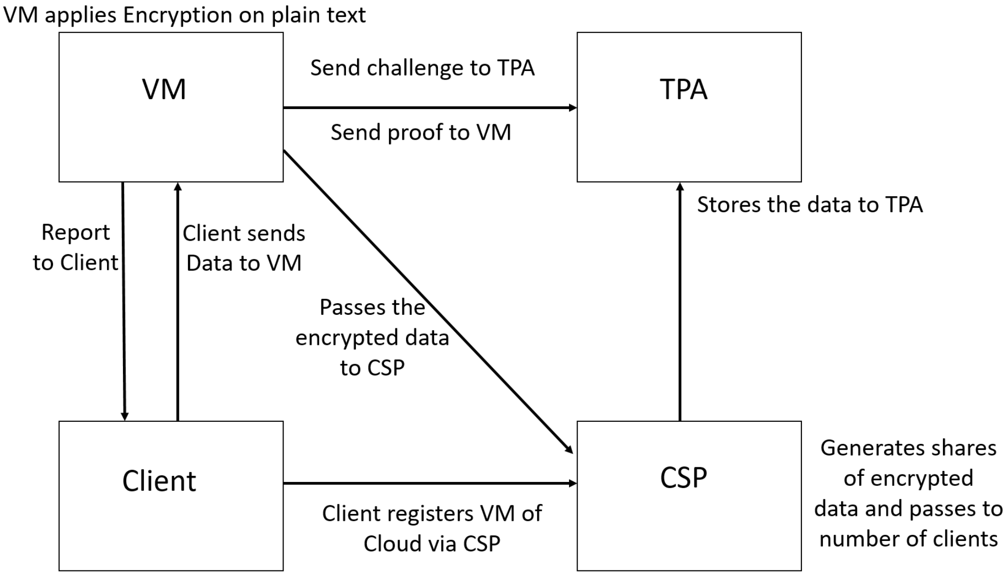

VM applies Encryption on plain text
VM
TPA
Send challenge to TPA
Send proof to VM
Report to Client
Client sends Data to VM
Passes the encrypted data to CSP
Stores the data to TPA
Client
CSP
Client registers VM of Cloud via CSP
Generates shares of encrypted data and passes to number of clients

Figure 2. Proposed model

(a) Client: Firstly client sends request to Cloud Service Provider (CSP) for granting access to virtual machine. Following the request from the client, the CSP acknowledges the client and grants access of the virtual machine of the cloud by providing user login and authentication code information. These secret information are generated randomly. After obtaining access, clients transfer plain text to Virtual machine.

(b) Virtual machine (VM):- At this stage, high level of security is required for the data stored in the cloud sent from the client. In spite of the CSP being trustworthy, an efficient key encryption method is applied for securing the data that is stored in the private area of cloud. After encryption is adopted, the encrypted stream of data is passed to the CSP.

(c) Cloud Service Provider: CSP passes the all data to TPA for storage. Here TPA is working as secured storage system by providing a transparent method for ensuring trust between the data owner (client) and the cloud server. This will not only help the clients to evaluate the risk of their subscribed service of the cloud server, but also will benefit the CSP to improve their cloud based service platform.

(d) TPA: Stores all encrypted data

6. SYSTEM FLOW OF PROPOSED ALGORITHM

In the above section we discussed the problem of integrity in cloud environment and proposed two different algorithms that are working as the basic model of the proposed system. We designed an optimized integrity verification algorithm with variant of both. We ensured effective security by converting the input message to corresponding hexadecimal form and applied efficient polynomial interpolation process. Our proposed module also ensures integrity enhancement and verification for each individual virtual machines. The advantage of this scheme can be viewed in terms of achieving more security because of use of random function, reducing the length of the code using Huffman technique and finally ensuring transfer of long message between communicating parties using simple polynomial interpolation calculations.

The proposed technique is working on increasing the strength of shares for secured distribution and construction. System is completely working on attack proof model. Strength of encryption data calculated is approximately 7.9. That is the maximum calculated results achieved till now. We present the flow of our proposed system in Figure 3.

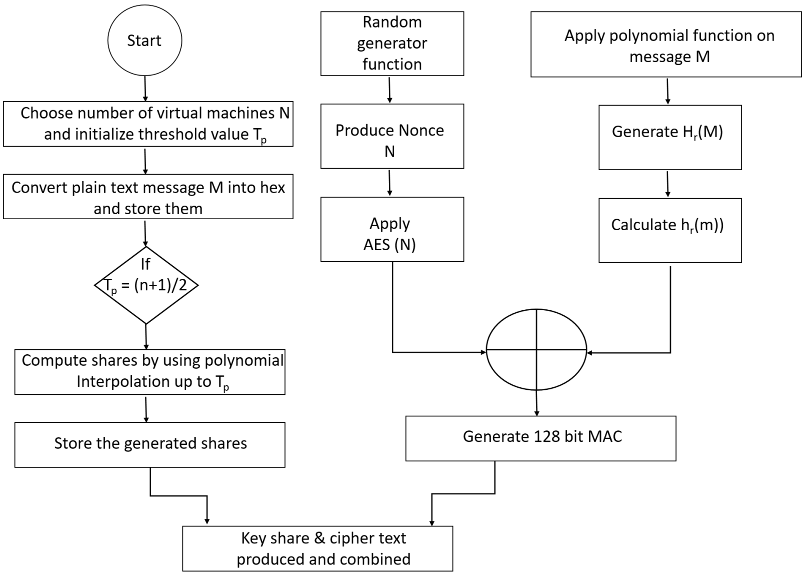
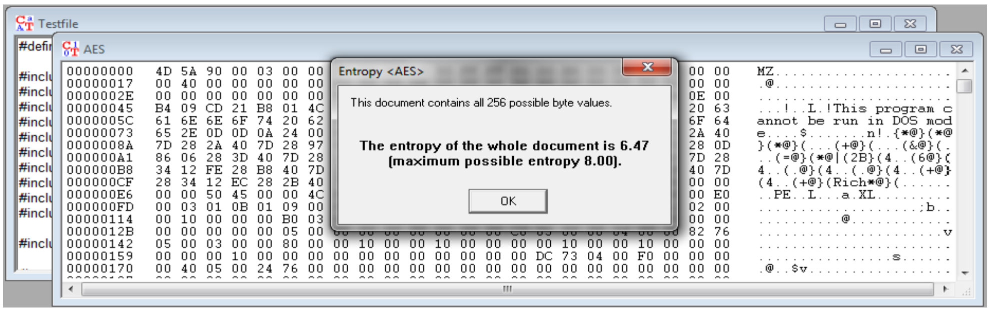

Start
Choose number of virtual machines N and initialize threshold value T_p
Convert plain text message M into hex and store them
If $T_p = (n+1)/2$
Compute shares by using polynomial Interpolation up to T_p
Store the generated shares
Random generator function
Produce Nonce N
Apply AES (N)
Apply polynomial function on message M
Generate $H_r(M)$
Calculate $h_r(m))$
Generate 128 bit MAC
Key share & cipher text produced and combined

Figure 3. Flow chart of proposed system

7. RESULTS AND DISCUSSION

In this section, we present the experimental results on our proposed system and discuss our findings accordingly. Here is the compilation of the basic or standard approach 6.47 out of 8 and in the designed optimized approach it's approximately 7.9. The screenshots of the entropy calculation by the standard AES and the polynomial AES are shown in Figure 4 and Figure 5 respectively.

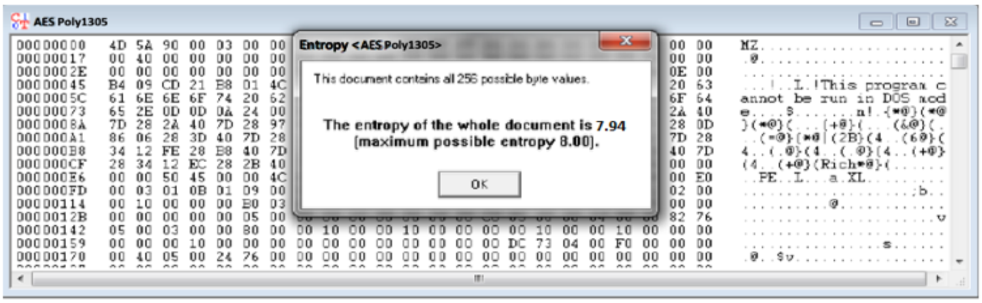


This document contains all 256 possible byte values.

The entropy of the whole document is 6.47
(maximum possible entropy 8.00).

```
00000000   4D 5A 90 00 03 00 00   00 00   MZ......................
00000017   00 40 00 00 00 00 00   00 00   .@......................
0000002E   00 00 00 00 00 00 00   0E 00   ........................
00000045   B4 09 CD 21 B8 01 4C   20 63   ...!..L.!This program c
0000005C   61 6E 6E 6F 74 20 62   6F 64   annot be run in DOS mod
00000073   65 2E 0D 0D 0A 24 00   2A 40   e....$.......n!.{*@}(*@
0000008A   7D 28 2A 40 7D 28 97   28 0D   }(*@}(...(+@}(...(&@}(.
000000A1   86 06 28 3D 40 7D 28   7D 28   ..(=@}(*@|(2B}(4..(6@}(
000000B8   34 12 FE 28 B8 40 7D   40 7D   4..(.@}(4..(.@}(4..(+@}
000000CF   28 34 12 EC 28 2B 40   00 00   (4..(+@}(Rich*@}(......
000000E6   00 00 50 45 00 00 4C   00 E0   ..PE..L...a.XL.........
000000FD   00 03 01 0B 01 09 00   02 00   ......................;b..
00000114   00 10 00 00 00 B0 03   00 00   ..........@............
0000012B   00 00 00 00 00 05 00   82 76   .......................v
00000142   05 00 03 00 00 80 00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00 00   ........................
00000159   00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 DC 73 04 00 F0 00 00 00   ..................s.....
00000170   00 40 05 00 24 76 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   .@..$v..................
```

Figure 4. AES entropy



This document contains all 256 possible byte values.

The entropy of the whole document is 7.94
(maximum possible entropy 8.00).

```
00000000   4D 5A 90 00 03 00 00   00 00   MZ......................
00000017   00 40 00 00 00 00 00   00 00   .@......................
0000002E   00 00 00 00 00 00 00   0E 00   ........................
00000045   B4 09 CD 21 B8 01 4C   20 63   ...!..L.!This program c
0000005C   61 6E 6E 6F 74 20 62   6F 64   annot be run in DOS mod
00000073   65 2E 0D 0D 0A 24 00   2A 40   e....$.......n!.{*@}(*@
0000008A   7D 28 2A 40 7D 28 97   28 0D   }(*@}(...(+@}(...(&@}(.
000000A1   86 06 28 3D 40 7D 28   7D 28   ..(=@}(*@|(2B}(4..(6@}(
000000B8   34 12 FE 28 B8 40 7D   40 7D   4..(.@}(4..(.@}(4..(+@}
000000CF   28 34 12 EC 28 2B 40   00 00   (4..(+@}(Rich*@}(......
000000E6   00 00 50 45 00 00 4C   00 E0   ..PE..L...a.XL.........
000000FD   00 03 01 0B 01 09 00   02 00   ......................;b..
00000114   00 10 00 00 00 B0 03   00 00   ..........@............
0000012B   00 00 00 00 00 05 00   82 76   .......................v
00000142   05 00 03 00 00 80 00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00 00   ........................
00000159   00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 DC 73 04 00 F0 00 00 00   ..................s.....
00000170   00 40 05 00 24 76 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   .@..$v..................
```

Figure 5. AES-poly1305 entropy

We present the comparison between the standard AES and the polynomial AES in Table 1. Our proposed system upgrade the entropy of AES with a variant of SSS in the cloud simulator environment. The standard AES algorithm showed entropy value of 6.47. In computing, a lack or lower of entropy can have a negative impact on performance and security. However, the proposed designed optimized approach gave entropy value of 7.94, which means improved performance and enhanced security in the cloud computing. Therefore, the AES-poly 1305 performance is improved by 22% than the standard AES algorithm.

Table 1. Comparisons of encryption algorithms

Encryption Algorithm	Entropy
AES	6.47
AES-Poly1305	7.94

8. CONCLUSION

In this paper, we presented a model that effectively address the issue of the data loss on cloud in case of dealing with storing personal data which is not transparent to users.We introduce a new entity in terms of a virtual machine that provides services and assurance beyond service level agreement (SLA). In the proposed model, a role of data handling and security is assured with association of third party auditor (TPA) by the virtual machine. We further demonstrate the applied technique for encryption, decryption and integrity verification modules. We also upgrade the entropy of the advanced encryption standard (AES) with a variant of secret sharing scheme in the environment of cloud simulator. Experimental results suggested that the proposed model has a great potential in addressing complicated security issues on the cloud.

REFERENCES

[1] R. Raghatate, S. Humne, and R. Wadhwe, "A survey on secure cloud computing using AES algorithm," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 12, pp. 295 – 301, 2014.

[2] D. Lowe and B. Galhotra, "An overview of pricing models for using cloud services with analysis on pay-per-use model," *International Journal of Engineering & Technology*, vol. 7, no. 3.12, 2018.

[3] G. Laatikainen, A. Ojala, and O. Mazhelis, "Cloud services pricing models," in *Software Business. From Physical Products to Software Services and Solutions*, 2013, pp. 117–129.

[4] W.-T. Tsai, X. Bai, and Y. Huang, "Software-as-a-service (saas): Perspectives and challenges," *Science China Information Sciences*, vol. 57, pp. 1–15, 05 2014.

[5] S. Pastore, "The platform as a service (paas) cloud model: Opportunity or complexity for a web developer?" *International Journal of Computer Applications*, vol. 81, pp. 29–37, 11 2013.

[6] T. Nodehi, S. Ghimire, and R. Jardim-Gonçalves, "Toward a unified intercloud interoperability conceptual model for iaas cloud service," in *2014 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, 2014, pp. 673–681.

[7] S. A. Hussain, M. Fatima, A. Saeed, I. Raza, and R. K. Shahzad, "Multilevel classification of security concerns in cloud computing," *Applied Computing and Informatics*, vol. 13, no. 1, pp. 57 – 65, 2017.

[8] Y. F. Shen, "The implementation of anti-attack AES mathematical model in library network encryption," in *Information Technology Applications in Industry, Computer Engineering and Materials Science*, vol. 756, 10 2013, pp. 2944–2947.

[9] D. Rountree and I. Castrillo, "Chapter 3 - cloud deployment models," in *The Basics of Cloud Computing*, 2014, pp. 35 – 47.

[10] M. Bogdanoski, P. Latkoski, A. Risteski, and B. Popovski, "IEEE 802.16 security issues: A survey," in *16th telecommunication Forum TELFOR*, 11 2008.

[11] J. Yuan and S. Yu, "Public integrity auditing for dynamic data sharing with multiuser modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1717–1726, 2015.

[12] T. Jiang, X. Chen, and J. Ma, "Public integrity auditing for shared dynamic cloud data with group user revocation," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2363–2373, 2016.

[13] S. Singh and S. Thokchom, "Public integrity auditing for shared dynamic cloud data," *Procedia Computer Science*, vol. 125, pp. 698 – 708, 2018, the 6th International Conference on Smart Computing and Communications.

[14] S. Singh, P. Sharma, and D. Arora, “Data integrity check in cloud computing using hash function,” *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, pp. 1974–1978, 2017.

[15] E. Meslhy, H. Abd elkader, and S. El-etriby, “Data security model for cloud computing,” *Journal of Communication and Computer*, vol. 10, pp. 1047–1062, 08 2013.

[16] A. Shamir, “How to share a secret,” *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[17] A. Maturu and K. T. R. Kumar, “An improved shamir secret sharing for secure communications,” *International Journal Of Engineering Science and Advanced Technology*, vol. 2, no. 6, p. 1608 – 1613, 2012.

[18] S. Gupta and S. Lamba, “An enhanced python based approach of secret sharing scheme with encryption,” *International Journal of Computer Science Engineering*, vol. 3, no. 3, pp. 173– 180, 05 2014.

[19] A. Muller, A. Ludwig, and B. Franczyk, “Data security in decentralized cloud systems — system comparison, requirements analysis and organizational levels,” *Journal of Cloud Computing*, vol. 6, no. 1, pp. 82:1–82:9, 2017.

[20] F. J. D’souza and D. Panchal, “Advanced encryption standard (AES) security enhancement using hybrid approach,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, 2017, pp. 647–652.

[21] A. Chandan, S. Gupta, RichaKumari, VrindaRajan, and S. Sonone, “Secure sharing of data in cloud computing with secret sharing algorithm,” *International Journal of Emerging Trend in Engineering and Basic Sciences*, vol. 2, no. 1, pp. 665–671, 2015.

[22] R. F. Olimid, “SETUP in secret sharing schemes using random values,” *Security and Communication Networks*, vol. 9, no. 18, pp. 6034–6041, 2016.

[23] J. L. Dautrich and C. V. Ravishankar, “Security limitations of using secret sharing for data outsourcing,” in *Data and Applications Security and Privacy XXVI*, N. Cuppens-Boulahia, F. Cuppens, and J. Garcia-Alfaro, Eds., 2012, pp. 145–160.

[24] D. J. Bernstein, “The poly1305-aes message-authentication code,” in *Fast Software Encryption*, H. Gilbert and H. Handschuh, Eds., 2005, pp. 32–49.

[25] R. SivaRanjani, L. Bhaskari, and P. Avadhani, “Secure message transmission using lagrange polynomial interpolation and huffman coding,” *International Journal of Computer Applications*, vol. 55, pp. 32–35, 10 2012.

BIOGRAPHIES OF AUTHORS

Iqbal Ahmed got his Bachelor of Science (BSc) Honors degree in Computer Science and Engineering from University of Chittagong, Bangladesh in 2007 and achieved joint Master degree from PERCCOM program of European Union in September 2015. He received his Master of Complex System Engineering degree from University of Lorraine (UL), France then Master in Technology from Lappeenranta University of Technology (LUT), Finland and Master degree in Pervasive Computing and Communication for Sustainable development from Lulea University of Technology (LTU), Sweden. He received his Ph.D. degree from the Department of Information Science, Saga University, Japan in 2018. He has been working as an Associate Professor in the department of computer science & Engineering in University of Chittagong since August 2018.

His current research interest lies in the field of green and sustainable computing, cloud computing and information processing.

He has been awarded Cat-A scholarship of Erasmus Mundus from European Union two times in 2010 and 2013 respectively.

Fahim Alam received the B.S.(Hons.) degree in computer science & engineering from University of Chittagong, Bangladesh, in 2008. He received the Masters degree in computer science from St. Francis Xavier University, Canada in 2012 and the Ph.D. degree from the Griffith University, Australia in 2019. He is currently working as an Assistant Professor in the department of computer science & Engineering in University of Chittagong.

His research interest includes pattern recognition and machine learning, computer vision and hyperspectral image analysis in the fields of remote sensing.

He was affiliated with IEEE as student member from 2014 to 2018. He has served as invited reviewer for several journals including IEEE Transactions of Geoscience and Remote Sensing.